LOVEGROVE MATHEMATICAL SERVICES

GREAT LIKELINESSES

(Versions B06 & B06C)

USER MANUAL

R.Lovegrove

GREAT LIKELINESSES is a program for calculating likelinesses, that is for finding best-estimates of probabilities.

The recommended citation for this manual is

Lovegrove,R.,(2013),'Great Likelinesses (versions B06 & B06C) User Manual', Lovegrove Mathematical Services, London, December 2013

London
United Kingdom
December 2013

www.lovegrovemaths.co.uk
loveroger1024@gmail.com

Contents

Todo list

1 Introduction

GREAT LIKELINESSES (the name is a pun on that of Charles Dickens's novel *Great Expectations*) is a program for calculating likelinesses, that is for finding best-estimates of probabilities.

This is not a textbook about the theory of likelinesses. It is a guide to the use of the program *Great Likelinesses.* A summary of the notation and terminology is given in Appendix A; the algorithms are outlined in Appendix B.

2 There is no warranty

This program is currently under development, and is not intended to be used in any situation where there are or might be deleterious consequences arising from that use.

There is no warranty of any form. For example, there is no warranty against failure of the program, or against failure of the program to produce the correct result.

By using the program, you accept full responsibility for all the consequences of that use. If you are not willing to accept that responsibility then do not use the program.

3 Versions

The program currently has two versions: B06, which is the full version, and B06C, which is the classical version.

In the full version, the degree is limited to a maximum of 77. In the classical version, the degree is restricted to values which have some significance in the classical problems to do with coins, dice and cards; these are 2,3,4,5,6,8,13,16,26,32,36,39,52 and 64.

4 Known Issues

- The clock doesn't work properly in the countdown. This is a display matter only, and has no bearing on the running of the program itself.

5 Basics

Basic concepts

A coin is of degree 2; a die is of degree 6; a pack of cards is of degree 52. *The degree is the number of possibilities that something (tossing a coin; rolling a die; drawing a card) may take.* It is the number of classes in a classification system.

The possibilities/classes are labelled $1, 2, \ldots, N$ where N is the degree.

Given a degree, we can always define a *distribution* with that degree, as in Table 1. For a distribution f, each $f(i) > 0$ and $\sum_{i=1}^{N} f(i) = 1$.

i	**1**	**2**	**3**	**4**	**5**	**6**
f(i)	0.2	0.1	0.35	0.05	0.25	0.5

Table 1: Distribution of degree 6

Likewise, we can define an *histogram* of degree N, as in Table 2. Each h(i)$\geq$ 0 and the h(i) are not required to be integer-valued. If each h(i) is an integer then h is called an *integram* (Table 3); an integram is just a special type of histogram. There are circumstances where an integram, rather than just an histogram, is required.

i	**1**	**2**	**3**	**4**	**5**	**6**
h(i)	3.0	4.2	0	0.7	1.8	0

Table 2: Histogram of degree 6

i	**1**	**2**	**3**	**4**	**5**	**6**
h(i)	3	4	0	0	1	0

Table 3: Integram of degree 6

Note that an histogram may take values of 0, but a distribution may not. The most important histogram is the zero integram, $\underline{0} = (0, 0, \ldots, 0)$.

If f is a distribution and g is an integram, both of degree N, then the probability of g given f is $Pr(g|f) = M(g)f^g$, where M(g) is the multinomial coefficient associated with g (see Appendix A), and $f^g = f(1)^{g(1)} \ldots f(N)^{g(N)}$.

The best-estimate of something is its mean value. In order to have a mean value, there must be a set of values to find the mean of. This program best-estimates $Pr(g|f)$ so there has to be a set of $Pr(g|f)$ to find the mean of. We start with a set of distributions, called the *Underlying Set*, calculate $Pr(g|f)$ for each f in that set and then find the mean of *those*.

The mean used is a weighted mean, where the weights depend upon available data in the form of an histogram (called the *given histogram*). The actual formula used is

$$L_P\left(g|h\right) = M\left(g\right)\frac{\int\limits_{f\in P} f^g f^h}{\int\limits_{f\in P} f^h} \tag{1}$$

where P is the underlying set, h is the given histogram and $L_P(g|h)$ is the likeliness (see Appendix A).

The expression for $M(g)$ contains the product of factorials, $\prod\limits_{i=1}^{N} g(i)!$ (see Appendix A), so each g(i) has to be integer-valued, so g has to be an integram. The weights, f^h, started life as $Pr(h|f)$, that is $M(h)f^h$, but the $M(h)$ has cancelled since it appeared in both the numerator and denominator of (1). The cancellation of the $M(h)$ has taken with it any need for h to be integer-valued so h may be an histogram rather than specifically an integram.

In a few simple cases, the main ones of which are listed in Appendix C, the integrals on the RHS can be evaluated theoretically.

Usually, however, a numerical approach is needed; that is what this program does. The process is very simple: we replace the integrals by summations and the underlying set, P, by a sample of points (ie. distributions) selected at random from P. That sample can be surprisingly large: about a million points are often needed -the program defaults to 750,000- but 100 million or more can at times be necessary. It is only recently that improvements in computer technology have made it possible for such large problems to be tackled on home computers.

What the program does

The program:-

- Finds $L_P(g|h)$, and other standard likelinesses, by sampling from P.
- Keeps track of convergence as the sampling process proceeds.
- Produces a separate sample of distributions from P, and uses each as a generating distribution to generate simulated observations.
- Calculates the best-estimate of the probability (ie. the likeliness) that $Pr(g|f) < x_i$ for a selection of x_i equally spaced across some interval specified by the user. Likewise for $Pr(1|f), \ldots, Pr(N|f)$. This is the likeliness equivalent of building up a CDF.
- For each of the subintervals $[x_i, x_{i+1}]$ calculates the Likeliness that $Pr(g|f)$ lies in that subinterval. This is the likeliness equivalent of building up a PDF.

It does this for an underlying set, P, chosen by the user from those listed in 10.2

In addition, the program can contract the underlying set onto any centre. It can also handle merged data-blocks.

Files used

The files used are given in Table 4.

Filename	Purpose
data.txt	Keeps details of problem, for future editing and/or use
results.csv	Results, for viewing in spreadsheet
defaults.txt	Various defaults, to personalise the program
sampling_dis.csv	Sample of distributions
sampling_obs.csv	Observations simulated by using the distributions in sampling_dis.csv
sampling_rfs.csv	Relative frequencies for the observations in sampling_obs.csv
errlog.txt	Keeps track of run-time errors
scratch1.txt, scratch2.csv	Scratchpads for the program's own use.

Table 4: Files produced by program

data.txt and results.csv

When you start a new problem, you type it in from the keyboard. Details are saved in the file data.txt so that you do not have to type them in again on subsequent runs.

Results are sent to the file results.csv for viewing in a spreadsheet.

If you want to keep data.txt or results.csv then make a copy, under a different name, in the usual way.

Countdown

While the program is carrying out an analysis, a countdown-to-completion is sent to the screen so that you can see progress. The analysis is in two parts: a fast initial pass, during which various items are roughly estimated so as to improve the efficiency of the program, followed by a slower second pass during which likelinesses are found. During the second pass, the countdown includes details of the estimated run-time: these estimates will be thrown out if the computer is used for any other purpose while the program is running.

Running from Windows

It is recommended that you run the program from within Windows rather than by switching, firstly, to DOS. This is because the program can be so fast with simpler analyses that the screen buffer cannot keep up and so forces the program to slow down significantly. Windows has improved screen-buffering which largely overcomes this.

6 Using Spreadsheets

The .csv files have been designed to be viewed within a spreadsheet.

Open your spreadsheet by right clicking on the icon for the file you want to open and selecting 'open with'. You should then be offered the choice of programs to use to open the file. Choose your favourite spreadsheet: you should then be offered a choice of options defining how the spreadsheet is to interpret the file.

Microsoft Excel

As the separator, choose a comma (,). As the text delimiter, choose a double quote ("). Do not choose to merge successive delimiters.

OpenOffice/Apache OpenOffice Calc

As the separator, choose a comma (,). As the text delimiter, choose a double quote ("). Do not choose to merge successive delimiters. For versions 3.3 and later of Calc, select 'detect special numbers' (you will not be offered this option in earlier versions).

LibreOffice Calc

As the separator, choose a comma (,). As the text delimiter, choose a double quote ("). Do not choose to merge successive delimiters. Select 'detect special numbers'.

Lotus 1-2-3

Choose 'Parse as CSV'. When the spreadsheet opens, it may seem that some of the fields have been asterisked out: they have not -it's just that the default column widths are too small.

In all spreadsheets, when viewing results.csv, you might find it helpful to widen Columns A and B.

7 What you should do now

1. Form a new folder to contain the files used by this program.
2. Transfer the .exe file and these notes into that folder.
3. Run the .exe file, and select item 999 from the opening menu. This will set up various defaults.

You can now experiment with the program, to get a feel for what's going on.

8 Running the program

To start the program, run the .exe file.

The opening screen gives you various options. If this is the first time that you have ever run this version then you must choose option 999 in order to set up various defaults.

Otherwise, your choice will normally be between Option 1 if you are starting a completely new problem, or Option 2 if you are repeating a previous problem or running a modification of one.

To start a new problem, choose Option 1. You will then be asked a number of questions, the answers to all of which (apart from the Title and Subtitle) will be numerical. For most of these, you will have standard answers which you will soon get used to giving very quickly; with practice, your fingers will type most of the answers faster than you think of them.

If you choose Option 2 then the computer will just take over and run the problem, giving you an on-screen progress report -which, for simpler problems, might flash past so quickly that you are unable to read it.

9 Stopping

The program will normally come to a stop of its own accord. There are times, though, when you might want to stop it prematurely.

You can choose ‘Stop’ from the opening menu. This will stop the program before it has really started.

During data-input, when asked to answer ‘1 for YES or 2 for NO’ give one of the emergency numbers 911 or 999. The program will interpret this as a request to stop. However, data.txt will be partially written and will be unuseable; the next time you run the program, you will have to input the whole of the problem from the keyboard.

Once data-input has finished, all you can really do is force a hard stop. You could switch the computer off and remove its batteries, but an easier way would be to press CTRL-C. The program will be forced to stop immediately, so it will not have the opportunity to close any open files. As a result, some files might not be closed properly –which could mean that you will need to restart your computer before you can use the program again. If you are analysing confidential or sensitive data then you should delete scratch1.txt and scratch2.csv manually.

10 The underlying set

10.1 Components of the underlying set

As a general rule, anything other than data (values of h(i) and g(i)) which has to be stated in order to define the problem forms part of the definition of the underlying set. Components include:-

- fundamental type
- degree
- contraction
- essential domains of modal distributions
- range of any mode
- whether modal distributions are bell-shaped

10.2 Fundamental types of underlying set

10.2.1 Unstructured, S(N)

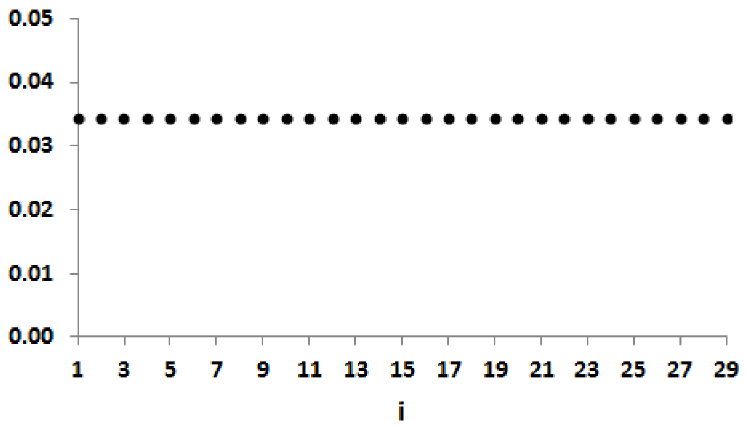


Figure 1: $L_{S(29)}(''i'')$

S(N) is the set of all distributions of degree N.

There is no relationship between the f(i) other than the requirement that they sum to 1. The fact that the domain, X_N, is ordered is irrelevant.

10.2.2 Ranked, R(N)

A distribution, f, is ranked if $f(1) > f(2) > ... > f(N)$.

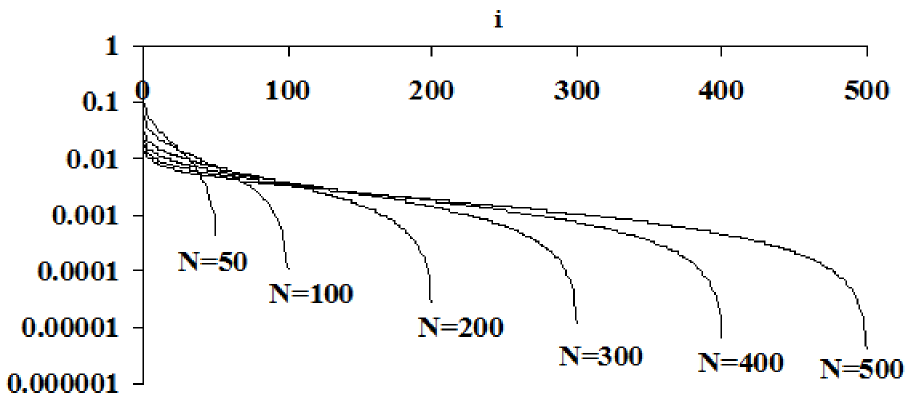
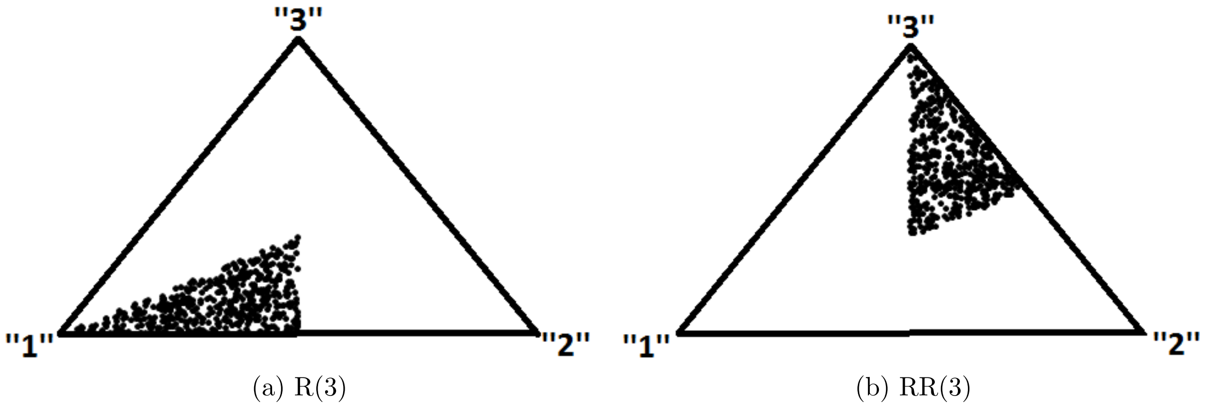


Figure 2: $L_{R(N)}(''i'')$ for various N

10.2.3 Reverse-ranked, RR(N)

The mirror-images of ranked distributions, these increase to the right:
$f(1) < f(2) \cdots < f(N)$.



(a) R(3) (b) RR(3)

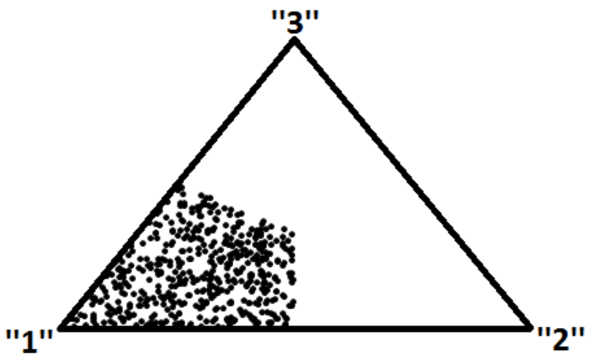
Figure 3: Positions of R(3) and RR(3) in S(3)

10.2.4 Stepdown, SD(c,N)

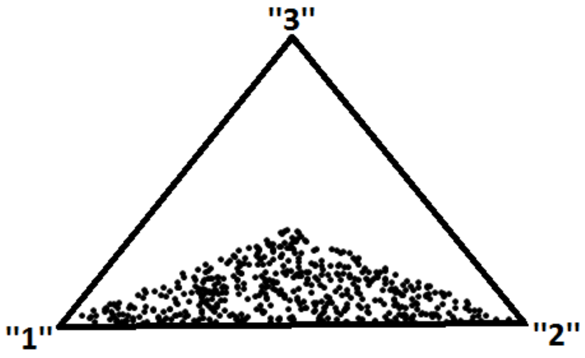
There is a c such that the function values below c are all greater than the function values above c; that is, $i \leq c < k \Rightarrow f(i) > f(k)$.

10.2.5 Ranked step-down RSD(c,N)

The same as stepdown except that, in addition, the functions values below c are ranked; that is $i < j \leq c < k \Rightarrow f(i) > f(j) > f(k)$. c can be interpreted as a limit of discrimination for a ranked distribution.

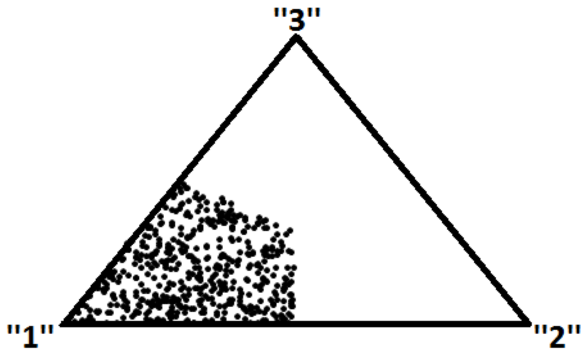


(a) SD(1,3)

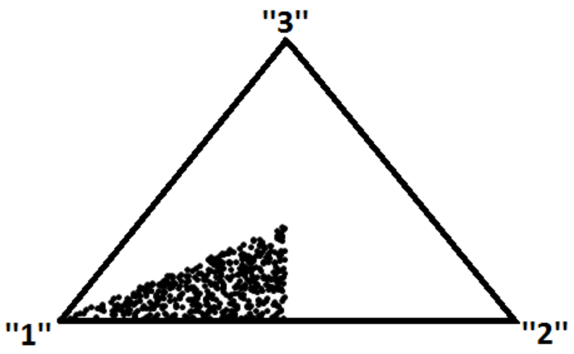


(b) SD(2,3)

Figure 4: Positions of SD(1,3) and SD(2,3) in S(3)

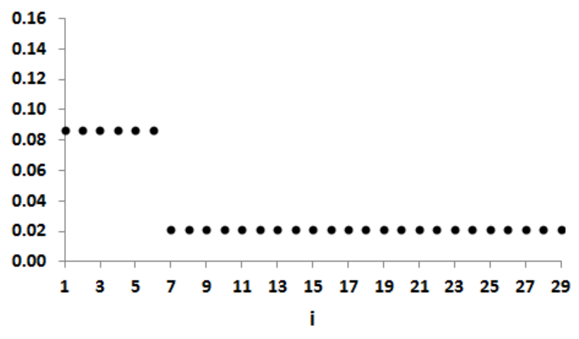


(a) RSD(1,3)

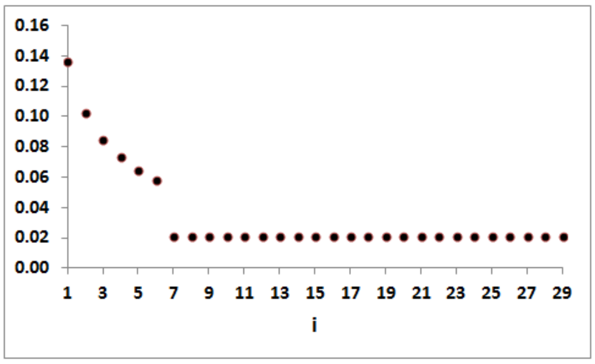


(b) RSD(2,3)

Figure 5: Positions of RSD(1,3) and RSD(2,3) in S(3)



(a) $L_{SD(6,29)}(''i'')$

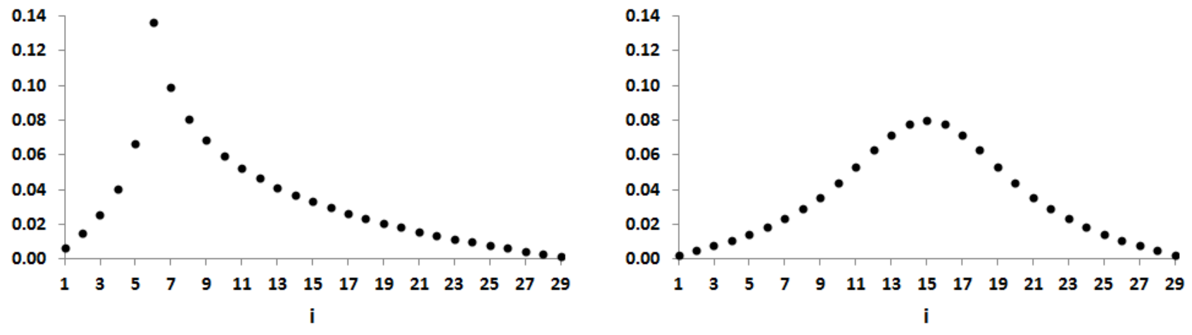


(b) $L_{RSD(6,29)}(''i'')$

Figure 6: Likelinesses for step distributions

10.2.6 Unimodal, M(A to B,N)

Each distribution has precisely one mode. The mode does not have to be the same for every distribution used in the analysis, although it can be if so required. The program asks for a range of i which the mode may take; the smallest value is A and the largest is B. You will usually use one of two specific cases, here.



(a) $L_{M(6,29)}(''i'')$ (b) $L_{M(29)}(''i'')$

Figure 7: Likelinesses for unimodal distributions

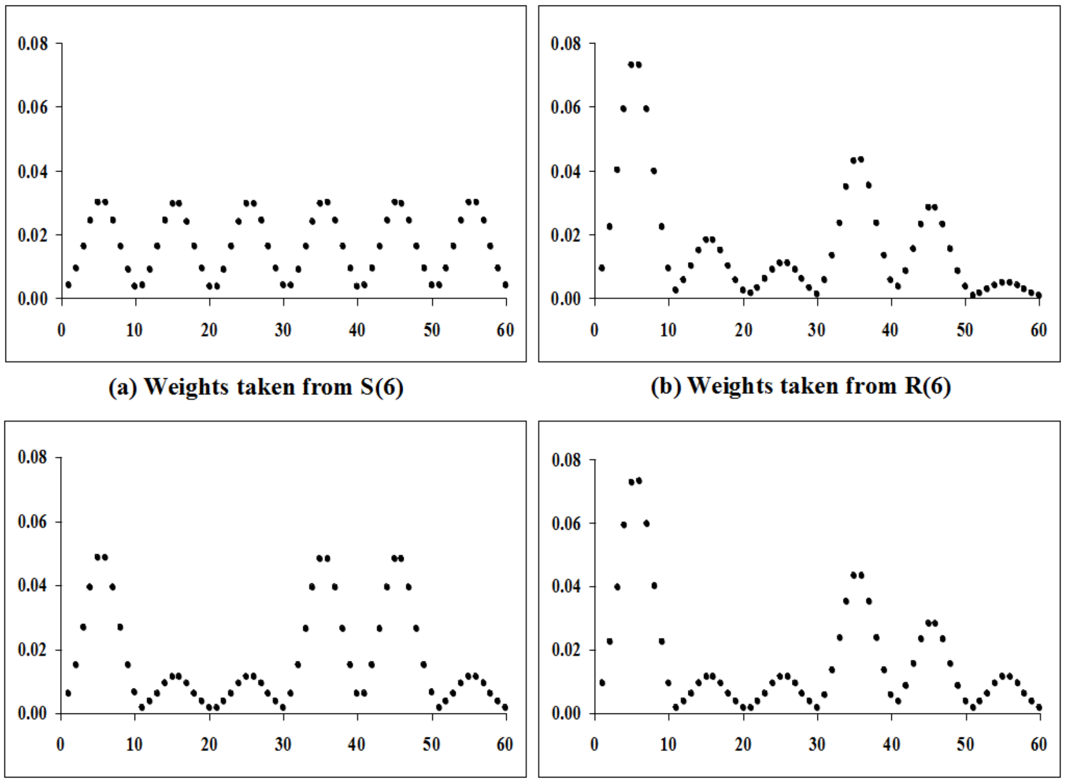
1. To use a specific mode, m say (Figure 7a has m=6), choose A=B=m. We then write $M(m, N)$ rather than M(m to m,N).
2. If $A = 1$ and $B = N$ then M(A to B,N) becomes M(1 to N,N), which is written as M(N) and is the set of all unimodal distributions of degree N (Figure 7b).

Despite the impression given by some mathematical texts, unimodal distributions are not usually bell-shaped (see Figure 7a): any bell-shape is due to vagueness in the mode (Figure 7b). You are offered the option to use only bell-shaped distributions, but you should not usually accept this offer.

10.2.7 Multi-modal

The domain, X_N, is covered by several 'mini' unimodal distributions, which may overlap. Each mini unimodal distribution has its own *essential domain*, which is extended to cover the whole of X_N by using values of zero elsewhere.

The Multi-modal distribution is then built up from those mini unimodal distributions by taking a linear combination of them. A linear combination needs weights: these are not specified precisely but are, rather, specified by giving a set from which they are to be selected; the program then selects them every time it is forming a distribution. The program labels the modes in the order in which their details are typed into the computer, which need not be left-to-right and so gives you flexibility when defining the problem. Figure 8 shows examples with weights selected from different sets, where the details of the modes were typed into the program in the order 1-4-5-2-3-6.



(a) Weights taken from S(6) (b) Weights taken from R(6)

(c) Weights taken from SD(3,6) (d) Weights taken from RSD(3,6)

Figure 8: Different types of weights

10.2.8 Reverse-ranked, RR(N)

A reverse-ranked distribution is the mirror image of a ranked distribution: values increase in size from left to right.

10.2.9 Reverse-ranked with unimodal slope

(As the name suggests)

10.2.10 Ranked with ranked slope

(As the name suggests)

10.2.11 Ranked with unimodal slope

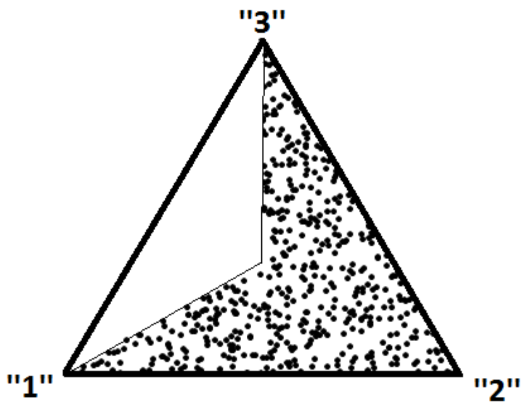
(As the name suggests)

11 Contractions

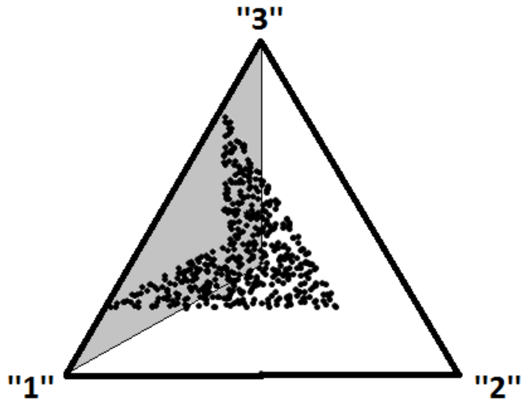
A contraction is a mapping of the form $f \mapsto q + \alpha(f - q)$. These are useful for reducing the size of the underlying set. α is the *magnitude*, and q the *centre*, of the contraction

Although you are given the opportunity just to type in the co-ordinates of the centre, if the degree is large then this can involve a lot of typing, which will be boring as well as increasing the chances of a typing mistake. The program therefore gives you the choice of several standard cases.

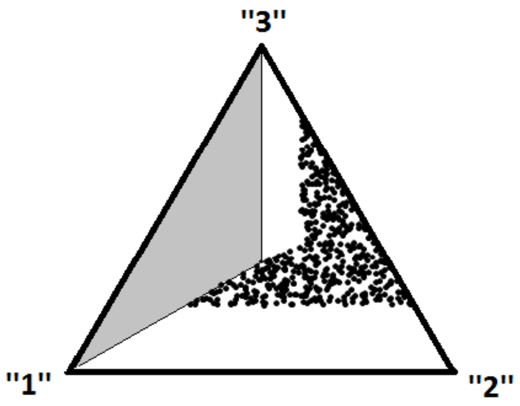
A contraction might cause the underlying set to fail to be of the same Fundamental type as originally chosen; for example, the contraction of an unimodal distribution might not be unimodal. Figure 9 shows M(3) together with two contractions with magnification 0.6. In Figures 9b & 9c, the shaded area, outside the boundary of M(3), shows the location of distributions which are not unimodal. The contraction in 9b causes a significant number of unimodal distributions to fail to be unimodal; however, every unimodal distribution remains unimodal under the contraction in 9c.



(a) $M(3)$



(b) Contraction centre $\frac{''3''+''1''}{2}$



(c) Contraction centre $\frac{''2''+''3''}{2}$

Figure 9: Two contractions of M(3) with magnification 0.6

12 The given histogram

12.1 Basics of the specification

The given histogram is specified in three parts, H1+H2+H3:-

1. H1 is specified as an histogram
2. H2 is specified as an ordered N'tuple of relative frequencies (which may be 0s) together with a sample size. The program finds H2 by multiplying the two together.
3. H3 is due to merge blocks, and will be discussed separately.

Because an histogram will usually consist mostly of zeroes, when giving H1 (also the relative frequencies for H2), you specify a block about which you want to be asked, and the program defaults values outside that block to zero.

To reduce errors, and also to make things easier for you, when you are specifying the relative frequencies for H2 you do not have to make them sum to 1: the program will normalise them for you.

12.2 Merge Blocks

Table 5 gives three examples of Merge blocks.

3 or fewer	4	5	6	7	8
9	2	0	3	3	1

(a) Merge block on left

1	2	3	4	5	6 or more
2	1	6	2	0	7

(b) Merge block on right

3 or fewer	4	5	6 or more
9	2	0	7

(c) Merge blocks at both ends

Table 5: Merge blocks

Each consists of a number of columns which have been merged together so that the detail has been lost of the entries in individual columns but the total of the entries is still known.

As shown in Table 5, Merge blocks can occur at either end of the table, or at both ends.

In practice, when data is collected in batches it sometimes happens that some batches contain a merge block but some do not. To cater for this, the program allows merge blocks to be entered independently of H1 and H2, and to overlap their non-zero entries.

The degree is the number of columns that there would have been had the merging not taken place.

The use of Merge Blocks introduces such vagueness into a problem that the resulting algorithms can be highly ill-conditioned. A significant increase in the number of iterations will be needed, but will not necessarily improve convergence to an acceptable extent; indeed, increasing the number of iterations could make convergence properties worse rather than better. For this reason, Merge Blocks should be used with caution.

If there is no Merge Block then H1+H2 is the given histogram. If there is a Merge Block, however, the situation is more complicated since the program uses the information about the Merge Block to recreate the third histogram, H3.

H3, however, is created every time that a new distribution is selected from the underlying set and will change as that distribution changes, so it is not possible to specify it. This continually-changing nature of H3 is a significant component of the vagueness which is introduced by the use of merge blocks.

13 The required integram

The required integram, g, is specified in the same way as is H1: by giving a block about which you want to be asked, with values outside that block defaulting to zero.

The calculated Likeliness of g is shown on-screen as part of the countdown display. This means that convergence can be monitored whilst the calculations are proceeding.

14 Frequency distributions and CDFs

The expected frequency distribution of $Pr(g|f)$ can be useful as a predictor of the distribution of relative frequencies. This will normally be concentrated in a fairly small region around $L_P(g|f)$, so it would be inefficient to construct it over the whole of [0,1]; instead, we use a smaller interval to cover the range of interest, and then partition that interval into cells.

The difficulty is that that range of interest cannot be chosen until the frequency distribution has been constructed, but the frequency distribution cannot be constructed until the range of interest has been chosen.

To get round this, the program carries out a quick first-pass through the iterations during which it collects sufficient information to enable it to make a reasonable first-estimate of the interval: which it then stores in data.txt. The intention is not to ‘get the interval right’, but, rather, to be able to present the user with enough information to make a better choice to suit his/her own needs, modifying data.csv accordingly.

To begin with, when choosing the intervals of interest you will not have the slightest idea which ranges to specify, so you will leave it to the program to make a choice. Sometimes it will get the range right; sometimes it will be totally wrong. Nonetheless, the program will usually provide you with enough information to make a better second guess, which you should use to edit data.csv before running the program again. With experiece, you will rarely need to run the program 3 times.

15 data.txt

You may edit data.txt with any simple text editor: use of a wordprocessor is not recommended. You might do this, for example, if you wanted to change the values of some data without having to retype the whole of the problem-specification.

There are two types of item in data.txt: structural and non-structural. Structural items affect the layout of the remainder of data.txt; non-structural items do not. You are strongly advised not to edit structural items because of the knock-on effects for the rest of the file (which are usually not as easy to predict as might be thought).

Each item is preceded by a brief description. Descriptions of non-structural items are in CAPITALS and are enclosed in square brackets [ ].

To help you find your way around, items in a block of similar items are usually preceded by an indication of where you are, eg 'h(11)'. These are not descriptions so the lower case and the round brackets should not be taken as indicating a structural item: the description is at the start of the block.

If you make a syntactical mistake whilst typing the details of a new problem then the program can, and will, ask you to re-enter the information. If you make a syntactical mistake when editing data.txt, however, then the program cannot ask you to re-enter the information, because the program will not be running. The first you will know of the mistake is when you subsequently try running the program and a run-time error occurs; details of this will be sent to the screen and to the file errlog.txt. Behaviour is similar to that of a compiler: the error might not be picked up immediately and the reported form of the error might not be the actual form.

The contents of data.txt are in a standard layout chosen to make subsequent editing as easy as possible, and so are not simply a repetition of your typing. The basic idea is that it is easier -and less error-prone- to alter an existing value than it is to insert an omitted value, so everything is specifically given and nothing is implied. Examples are:-

> – When specifying an histogram, you give a block about which you wish to be asked, and the program defaults values outside that block to zero. Regardless of which block you specify, the block stored in data.txt always runs from 1 to N and the defaulted zero values are all specifically given.

– When specifying the subintervals to be used for the calculation of cdfs and pdfs, you are asked whether you want to specify them yourself or whether you want to leave that to the program. Regardless of how you reply, data.txt always contains the answer ‘do it myself’ followed by the intervals the program has chosen or were selected by you.

– Regardless of whether or not you say that you want to use a contraction, the program always gives you one, albeit one which has no effect because it has a magnitude of 1. So the answer to the question ‘Do you want to use a contraction?’ is always stored as ‘1’ for ‘YES’, followed by the centre and magnitude of a contraction. If you say that you do want a contraction then the stored details will be as specified by you. If you say that you do not want a contraction then the magnitude will be 1 (and you don’t have to worry about where the centre is, because the contraction will have no effect).

– When specifying the centre of a contraction, you are given the choice between various standard cases and specifying all the co-ordinates yourself. Regardless of how you reply, data.txt contains the answer ’specify them myself’, followed by all the co-ordinates.

The two things you will most often want to alter are (a) the number of iterations and (b) the subintervals used for the calculation of distributions. For convenience, these have been placed together, and are preceded by a line of asterisks across the screen, terminating with the words FREQUENT CHANGES HERE.

16 results.csv

results.csv is in the form of various tables, as below.

Table 1: Input Data

This Table summarises the data as input by you.

If you are not using any Merge blocks, then each will be shown as ‘not used’. For any Merge block that you are using, the column ‘size’ will show the number of observations you have specified for that block, and the extent of the block will be shown by the fields that have been ‘asterisked out’.

If you are not using a contraction then the row ‘contraction’ will show a contraction of size 1 and centre ″1″. If you are using a contraction, then its size and centre will be as specified by you.

Table 2: Random selection of 25 distributions

For convenience, either for your own interest or for use when writing a presentation, this section shows a random selection of 25 distributions. Also shown is $Pr(g|f)$.

Table 3: Convergence of Likelinesses

This table shows convergence of the calculated likelinesses as the iterations proceed. Plotted points are concentrated towards the beginning and end of the iterative process.

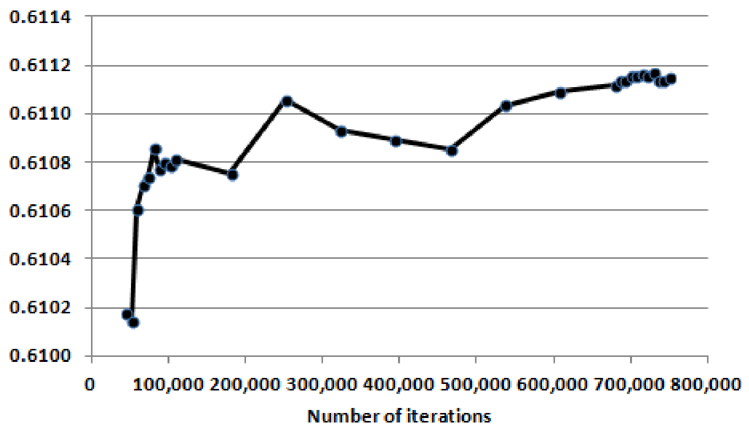


Figure 10: Example of the convergence of calculations

This table gives a very good indication of whether or not enough iterations had been used; it should always be checked, as a matter of routine.

Table 4: Likelinesses

This will be the main table of interest: it gives the likelinesses of the standard integrams (the required integram, g, and each of the integrams $''i''$).

It also shows the Multinomial Consistency, which is an indication of how well the likelinesses obey the Multinomial Theorem. This will usually be just for the purposes of reporting, since the program itself makes use of the Multinomial Theorem unnecessary.

Tables 5 & 6: Frequency and Cumulative distributions

The program partitions each range of interest (see Section 14) into 20 subintervals and finds the likeliness that $Pr(g|f)$ or $Pr(i|f)$, as appropriate, is in each of those subintervals. The results are output as Table 5, which consists of a number of small tables, one for each of the standard integrams. The centres of the cells have been included to make graph-plotting easier.

The cumulative sums of the likelinesses in each of the sub-tables of Table 5 are then formed, to give the best-estimated CDF for $Pr(g|f)$ and for each $Pr(i|f)$. These are output as Table 6.

17 The Sampling files

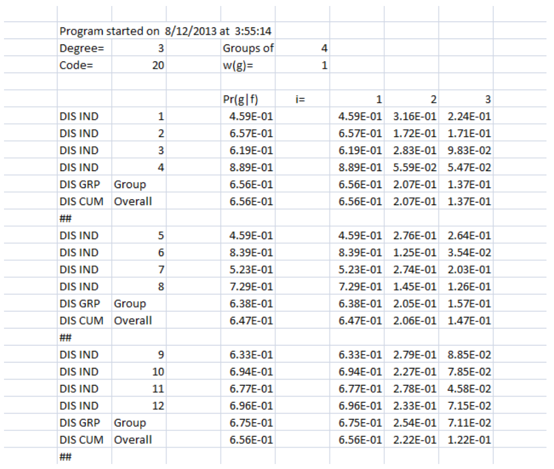
Every time the program is run, it forms three *sampling files*, as in Table 6.

Filename	**Content**
sampling_dis.csv	Sample of distributions
sampling_obs.csv	Observations simulated by using the distributions in sampling_dis.csv as generating distributions.
sampling_rfs.csv	Relative frequencies for the observations in sampling_obs.csv

Table 6: The sampling files

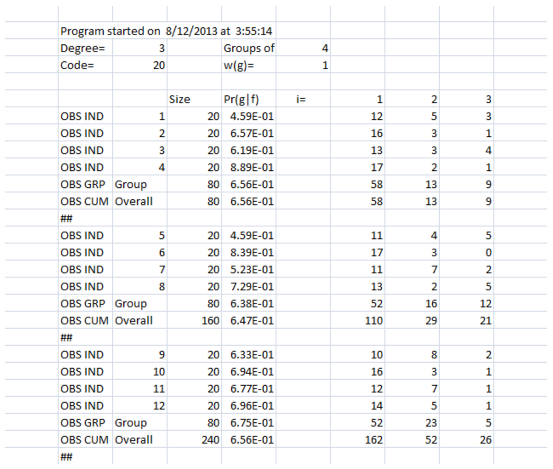
How many distributions there are in the sample, and how many simulated observations there are per distribution, are controlled by altering the appropriate values in the file defaults.txt.

The distributions may also be split into groups (Figure 11), the size of which is also set in defaults.txt.



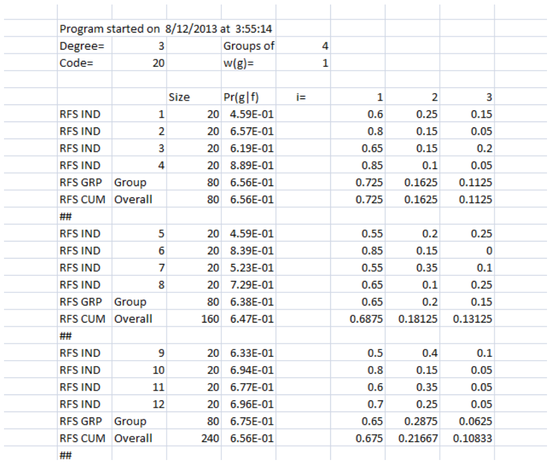
Program started on 8/12/2013 at 3:55:14							
Degree=	3		Groups of	4			
Code=	20		w(g)=	1			
			Pr(g|f)	i=	1	2	3
DIS IND	1		4.59E-01		4.59E-01	3.16E-01	2.24E-01
DIS IND	2		6.57E-01		6.57E-01	1.72E-01	1.71E-01
DIS IND	3		6.19E-01		6.19E-01	2.83E-01	9.83E-02
DIS IND	4		8.89E-01		8.89E-01	5.59E-02	5.47E-02
DIS GRP	Group		6.56E-01		6.56E-01	2.07E-01	1.37E-01
DIS CUM	Overall		6.56E-01		6.56E-01	2.07E-01	1.37E-01
##							
DIS IND	5		4.59E-01		4.59E-01	2.76E-01	2.64E-01
DIS IND	6		8.39E-01		8.39E-01	1.25E-01	3.54E-02
DIS IND	7		5.23E-01		5.23E-01	2.74E-01	2.03E-01
DIS IND	8		7.29E-01		7.29E-01	1.45E-01	1.26E-01
DIS GRP	Group		6.38E-01		6.38E-01	2.05E-01	1.57E-01
DIS CUM	Overall		6.47E-01		6.47E-01	2.06E-01	1.47E-01
##							
DIS IND	9		6.33E-01		6.33E-01	2.79E-01	8.85E-02
DIS IND	10		6.94E-01		6.94E-01	2.27E-01	7.85E-02
DIS IND	11		6.77E-01		6.77E-01	2.78E-01	4.58E-02
DIS IND	12		6.96E-01		6.96E-01	2.33E-01	7.15E-02
DIS GRP	Group		6.75E-01		6.75E-01	2.54E-01	7.11E-02
DIS CUM	Overall		6.56E-01		6.56E-01	2.22E-01	1.22E-01
##							

(a) sampling_dis.csv



Program started on 8/12/2013 at 3:55:14							
Degree=	3		Groups of	4			
Code=	20		w(g)=	1			
		Size	Pr(g|f)	i=	1	2	3
OBS IND	1	20	4.59E-01		12	5	3
OBS IND	2	20	6.57E-01		16	3	1
OBS IND	3	20	6.19E-01		13	3	4
OBS IND	4	20	8.89E-01		17	2	1
OBS GRP	Group	80	6.56E-01		58	13	9
OBS CUM	Overall	80	6.56E-01		58	13	9
##							
OBS IND	5	20	4.59E-01		11	4	5
OBS IND	6	20	8.39E-01		17	3	0
OBS IND	7	20	5.23E-01		11	7	2
OBS IND	8	20	7.29E-01		13	2	5
OBS GRP	Group	80	6.38E-01		52	16	12
OBS CUM	Overall	160	6.47E-01		110	29	21
##							
OBS IND	9	20	6.33E-01		10	8	2
OBS IND	10	20	6.94E-01		16	3	1
OBS IND	11	20	6.77E-01		12	7	1
OBS IND	12	20	6.96E-01		14	5	1
OBS GRP	Group	80	6.75E-01		52	23	5
OBS CUM	Overall	240	6.56E-01		162	52	26
##							

(b) sampling_obs.csv



Program started on 8/12/2013 at 3:55:14							
Degree=	3		Groups of	4			
Code=	20		w(g)=	1			
		Size	Pr(g|f)	i=	1	2	3
RFS IND	1	20	4.59E-01		0.6	0.25	0.15
RFS IND	2	20	6.57E-01		0.8	0.15	0.05
RFS IND	3	20	6.19E-01		0.65	0.15	0.2
RFS IND	4	20	8.89E-01		0.85	0.1	0.05
RFS GRP	Group	80	6.56E-01		0.725	0.1625	0.1125
RFS CUM	Overall	80	6.56E-01		0.725	0.1625	0.1125
##							
RFS IND	5	20	4.59E-01		0.55	0.2	0.25
RFS IND	6	20	8.39E-01		0.85	0.15	0
RFS IND	7	20	5.23E-01		0.55	0.35	0.1
RFS IND	8	20	7.29E-01		0.65	0.1	0.25
RFS GRP	Group	80	6.38E-01		0.65	0.2	0.15
RFS CUM	Overall	160	6.47E-01		0.6875	0.18125	0.13125
##							
RFS IND	9	20	6.33E-01		0.5	0.4	0.1
RFS IND	10	20	6.94E-01		0.8	0.15	0.05
RFS IND	11	20	6.77E-01		0.6	0.35	0.05
RFS IND	12	20	6.96E-01		0.7	0.25	0.05
RFS GRP	Group	80	6.75E-01		0.65	0.2875	0.0625
RFS CUM	Overall	240	6.56E-01		0.675	0.21667	0.10833
##							

(c) sampling_rfs.csv

Figure 11: Contents of the Sampling files

The sampling files are useful in modelling and in the design of experiments. Think of each row as representing the results of a test, where each test consists of a number of observations. Each group represents the results of an experiment, where each experiment consists of a number of tests. The whole file consists of a number of repetitions of an experiment, and shows the variation which might be expected.

Many users will not have any need for the sampling files, but for those who do they can be the most valuable part of the program. If a sample of distributions is needed just to show what distributions look like, eg in a report, then it should be remembered that results.csv contains a sample of 25 distributions intended for that purpose.

Each line starts with a status field, which says what that line is all about. By sorting on this, a file can be split into its various components of individual results, group sums/averages and grand total.

In sampling_dis.csv, the Cumulative Sums show the means over all the completed groups; in sampling_obs.csv, the sums; in sampling_rfs.csv, the overall relative frequencies. Details from any incomplete group, at the end of the file, are not included.

The program does not place an upper limit on the number of distributions which may be sampled. However, a spreadsheet might: an older spreadsheet might have a maximum of about 65,000 rows. A more modern spreadsheet will normally cater for more than 1 million rows.

18 defaults.txt

defaults.txt is intended to contain the answers to questions which most people either would not be interested in, or would not usually want to alter. These questions could rapidly become annoying if asked every time the program was run.

Because defaults.txt is much simpler than data.txt, error-reporting is minimal: either something works or it produces a run-time error. If the latter then the cause of the problem is easily spotted: usually either a text string has not been enclosed in quotes or a non-integer numerical value has been used.

Each item has a Factory Setting, which is hardwired into the program. All items can be reset to their factory settings by selecting item 999 from the opening menu.

Default number of iterations to be used

Each iteration corresponds to one distribution selected at random from the underlying set. Specify the default number here.

It is possible to alter the number of iterations within the program, but the number you put here should –if chosen correctly for the types of problem you are normally involved with– save you from having to do so most of the time. Just be careful not to use any thousands separators etc.

If you rarely have an interest in anything apart from basic likelinesses then it should be possible to reduce the default number of iterations to substantially fewer than the Factory Setting: 100000 or fewer will often be good enough (for example, see Figure 10). However, if the program runs fast enough for you then you should ask yourself why you are reducing the number. On the other hand, if you are usually interested in PDFs then you might find that an increase to substantially more than the Factory Setting would be convenient.

Factory Setting: 750000

The remainder of defaults.csv is concerned with the sampling files.

Number of distributions wanted in sampling_dis.csv

Each time the program is run, a number of distributions meeting the problem-definition is sent to the file sampling_dis.csv. This item specifies how many there should be.

Factory Setting: 100

Broken into groups of

Every n'th distribution is followed by group-level figures. Insert the value of n here.

A group size of 0 or less, or of more than the number of distributions, forces a single group consisting of all the distributions.

In sampling_dis.csv, each group is followed by the group mean; in sampling_obs.csv, by the group sum; in sampling_rfs.csv, by the overall relative frequencies. These are then followed by similar figures for the whole of the sample to that point.

Factory Setting: 0

How frequently a new distribution is to be chosen

There are two options:-

1. every time a distribution is written to sampling_dis.csv
2. only at the start of each group.

Factory Setting: 1

Code giving the number of observations generated per distribution

Each distribution sent to sampling_dis.csv is used as a generating distribution to simulate at least one observation. You specify the actual number of observations here by giving an integer, n, which has the effect given by Table 7.

n	Number of observations simulated per distribution
1,2, ...	n
0	$\omega(g)$
-1	chosen at random from $\{1,\ldots,\omega\,(g)\}$
-2,-3, ...	chosen at random from {1, ..., -n}

Table 7: Specifying the number of observations to be simulated.

If n is negative, a random choice of the number of observations is made every time (ie. it is not a 'once-and-for-all' decision).

Factory Setting: 0

19 Odds and Ends

(An unordered list of things to remember and things which do not easily fit in elsewhere.)

- For basic problems (no merging, contractions or Relative Frequencies; no given data; required integram ="1"; 750,000 iterations.), run times using a 64-bit laptop were as given in Table 8.

Fundamental	Degree, N					
type	2	5	10	25	50	75
S(N)	2	3	5	13	32	60
R(N)	2	3	5	13	33	61
M(N)	2	3	7	18	47	89
U(N)	2	3	7	18	49	90

Table 8: Typical run times (seconds)

- If you have been looking at any file but have forgotten to close it down before running the program again then you will receive a run-time error or be thrown back into Windows. Close the file and –if your system offers you the choice– choose **R**etry. If it does not offer you this choice then you may need to restart your computer.
- When using a spreadsheet to plot results, pay careful attention to the scale of the vertical axis. Spreadsheets usually choose the scale so as to maximise the vertical spread of the plotted points: this can cause the results to seem highly scattered when they are in fact in agreement to several sig figs. See Figure 10 for an example.
- To investigate the effects of the sample size of the given data, take advantage of the fact that input relative frequencies are normalised before use, so they do not actually have to be relative frequencies provided they are not negative. Do not give any data as the input histogram but give it, instead, as input relative frequencies; varying the sample size then does just that.
- In defaults.csv, if you choose to have a single group, by eg. selecting a group size of 0, and also choose to have a new distribution only at the start of a group then every entry in the sampling files will use the same distribution. However, that distribution will be selected at random and you will not have any say in its choice.

  To have just a single distribution, ***specified by you***, throughout the whole of the sampling files, when running the program, specify a contraction of magnitude zero, centred on the required distribution.

20 Troubleshooting

The program freezes immediately after I have chosen Item 2 from the opening menu, leaving the menu on the screen

You probably stopped the program on the previous run by using one of the emergency numbers 911 or 999. These make data.txt unusable (see 9, page 8) so the program has frozen while trying to read from it.

You will firstly need to clear the computer by restarting it. Then open data.txt: if it starts with a message saying that it was formed when the previous run was finished early, then that is the problem. You will need to run the program from the keyboard.

The program freezes soon after I have selected Item 2 from the opening menu, leaving the message * downloading of samples now completed *** on the screen**
You probably forced the program to stop on the previous run. This makes data.txt unusable. Take the same action as for the previous problem.

I get a run-time error with the error number M6101

This is an under/over-flow problem. There are various possible causes:

1. The given histogram has too large a sample size, causing underflow. The definition of likelinesses involves the factor f^h (see Appendix A), so a large sample size can lead to very small numbers. The program has been written to handle numbers down to about 10^{-600}, but this is sometimes not small enough.

   If this should happen to you then the best you can probably do is reduce the sample size of the given data, by inputting it as relative frequencies and then reducing the sample size until you find one which works.

2. You have been experimenting with the program and have used a completely unrealistic example which has a large h(i) associated with a very unlikely i. This is equivalent to having a large number of observations of something that is very unlikely to happen.

3. The required integram has too large a sample size. The calculation of M(g) involves several factorials (again, see Appendix A), which can quickly exceed the limits of double precision in the intermediate calculations even if the final value is within limits. The program has been written to minimise this problem but it cannot be completely avoided.

A Notation and Terminology

Let $\mathbb{R}^+$ be the non-negative reals, and $\mathbb{N}^+$ be the non-negative integers. For $N \in \mathbb{N}$ let $X_N = \{1, \ldots, N\}$. N is called the *degree.*

Let $G(N) = \{g | g : X_N \to \mathbb{N}^+\}$, $H(N) = \{h | h : X_N \to \mathbb{R}^+\}$, so $G(N) \subset H(N)$. The elements of H(N) are called *histograms* on X_N and those of G(N) *integer-valued histograms*, shortened to *integrams*, on X_N. The histogram h is identified with the point $(h(1), \ldots, h(N))$.

For $h \in H(N)$, the *sample size* of h is $\omega(h) = \sum\limits_{i=1}^{N} h(i)$.

For $n \in \mathbb{N}^+$, $\Omega_N(n) = \{g \in G(N) | \omega(g) = n\}$. This is the set of all integrams of degree N and sample size n. In particular, $\Omega_N(\omega(g))$ is the set of all integrams with the same sample size as g.

For $g \in G(N)$, the *Multinomial coefficient associated with g* is

$$M(g) = \frac{\omega(g)!}{\prod\limits_{i=1}^{N} g(i)!}.$$

Let $f : X_N \to]0,1]$ be such that $\sum\limits_{i=1}^{N} f(i) = 1$. Then f is called a *distribution on X_N* . $S(N)$ is the set of all such distributions. $S(N) \subset H(N)$.

For g∈G(N), h∈ H(N) and $P \subset S(N)$ where $P \neq \emptyset$, we define

$$L_P\left(g|h\right) = M\left(g\right) \frac{\int\limits_{f \in P} f^g f^h}{\int\limits_{f \in P} f^h}$$

where $\int$ is the Daniell integral.

$L_P\left(g|h\right)$ is called the *likeliness, over P, of g given h.* Since P, g or h will usually be clear from the context, this terminology is normally shortened by omitting appropriate terms.

h is called the *given histogram*, g the *required integram* and P the *underlying set.* More generally, any non-empty subset of S(N) is called *the underlying set* in S(N).

The integram of degree N and sample size 0 is $(0, \ldots, 0)$, which is denoted by $\underline{0}$, or –if greater clarity is needed– by $\underline{0}_N$. We have $L_P\left(\underline{0}|h\right) = 1$ for all (h,P). $L_P(g|\underline{0})$ is written as $L_P\left(g\right)$.

If we roll a die and throw the number 2 then we have not only thrown a 2 once but have also thrown 1, 3, 4, 5 and 6 zero times each. So we can think of ourselves as having thrown the integram (0,1,0,0,0,0). Also, we have not actually thrown the number 2 but have, rather, thrown the face labelled "2". It is very convenient to adopt notation which associates the symbol "2" with (0,1,0,0,0,0).

We define $''i''_N$ to be that integram $(x_1, \ldots, x_N)$ for which $x_i = 1$ but $x_n = 0$ otherwise; for example, $''2''_6 = (0,1,0,0,0,0)$. It is usually possible to write $''i''$ rather than $''i''_N$ without introducing ambiguity. Importantly, $f^{''i''} = f(i)$ and $M(''i'') = 1$.

If P is a singleton set, $P = \{f\}$, then $L_P\,(g|h) = M\,(g)\,f^g$, which is denoted by $Pr\,(g|f,h)$: since this is independent of h the notation may be simplified to $Pr\,(g|f)$; however, the presence of the h, although technically unnecessary, can sometimes add clarity.

Now let $V \subset S(N)$. Then the *likeliness of V, over P and given h*, is

$$L_P(V|h) = \frac{\int_{V\cap P} f^h}{\int_P f^h}.$$

For $x \in [0,1]$ let $V_x = \{f \in S(N)|Pr(g|f) < x\}$. Then $L_P(V_x|h)$ is the likeliness, over P and given h, of the set of those $f \in P$ for which $Pr(g|f) < x$. We denote this by $L_P(Pr(g|f) < x|h)$.

The function $[0,1] \to [0,1] : x \mapsto L_P(Pr(g|f) \leq x|h)$ is the *expected CDF of* $Pr(g|f)$.

Likewise, if $0 \leq x_0 \leq x_1 \leq 1$ then we define $L_P(Pr(g|f) \in [x_0,x_1]|h)$ to be $L_P(V|h)$ where $V = \{f \in P|Pr(g|f) \in [x_0,x_1]\}$. By covering [0,1] by cells in this way, we obtain an *expected frequency distribution for* $Pr(g|f)$.

B The Algorithms

B.1 The commoner underlying sets

The symbols representing the commoner underlying sets are given in Table 9

Symbol	Meaning
S(N)	The set of all distributions of degree N
R(N)	The set of all ranked distributions of degree N: $f(1) > ... > f(N)$
RR(N)	The set of all reverse-ranked distributions of degree N: $f(1) < \cdots < f(N)$
M(A to B,N)	The set of all unimodal distributions of degree N with mode between A & B inclusive
M(m,N)	M(m to m,N)
M(N)	M(1 to N,N)
SD(c,N)	For $c \in X_{N-1}$, the set of all step-down distributions of degree N with step at c: $i \leq c < k \Rightarrow f(i) > f(k)$
RSD(c,N)	The set of all ranked step-down distributions of degree N with step at c: $i < j \leq c < k \Rightarrow f(i) > f(j) > f(k)$

Table 9: The commoner underlying sets

B.2 Selection of $f \in S(N)$

Using the computer's RAND function, select $(N-1)$ points in $]0,1[$, and use them to partition $]0,1[$, resulting in N subintervals. Use the subinterval-lengths as the f(i), randomising them, first, to reduce any biase in the selection process.

B.3 Selection of $r \in R(N)$

$$\text{Let } A_N = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{N} \\ & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{N} \\ & & \frac{1}{3} & \cdots & \frac{1}{N} \\ & & & \cdots & \cdots \\ & & & & \frac{1}{N} \end{bmatrix}.$$

Then $\phi_N : S(N) \to R(N) : f \mapsto r$ where $r^T = A_N f^T$ is a linear bijection.

Select $f \in S(N)$ and then, starting with $r(N) = \frac{1}{N} f(N)$ and working upwards, construct $r = \phi_N(f)$.

B.4 Selection of $f \in RR(N)$

Select $r \in R(N)$ and then set $f(i) = r(N+1-i)$.

B.5 Selection of $f \in RSD(c, N)$

Select $r \in R(N)$ and randomise $\{r(c+1), \ldots, r(N)\}$.

B.6 Selection of $f \in SD(c, N)$

Select $f \in RSD(c, N)$ and randomise $\{f(1), \ldots, f(c)\}$.

B.7 Selection of $f \in M(m, N)$

The set of all injective unimodal distributions of degree N which have a mode of m is denoted by M(m,N). The practical underlying set, M(A to B,N) is formed as the union of $M(A, N), \ldots, M(B, N)$. [The set of non-injective unimodal distributions has measure zero.]

The basic procedure for forming $f \in M(m, N)$ is to select $r \in R(N)$ and then permute the r(i) to produce an unimodal distribution with mode m. The algorithm needs to determine how that permutation is to be carried out.

By considering the selection of (m-1) values to the left of m out of the (N-1) available (since $r(1) \mapsto m$), it follows that there are ${}^{N-1}C_{m-1}$ unimodal permutations of r which have a mode at m.

Place r(1), then r(2), then r(3) etc as follows:-

1. Since it must be that $r(1) \mapsto m$, place r(1) at m.
2. Since the distribution is to be unimodal, r(2) must be placed at either (m-1) or (m+1). In general, at each stage, the already-placed r(i)s must form a contiguous block, with the next value being placed at either end; we need to choose which end.

   Let there be L unfilled places to the left of the block and R unfilled places to the right. Then, of the ${}^{L+R}C_L$ possible ways in which the remaining (L+R) values may be placed, the number which have the next value to the left is ${}^{L+R-1}C_{L-1}$, so the proportion which have the next value to the left is the ratio of ${}^{L+R-1}C_{L-1}$ to ${}^{L+R}C_L$, which is $\dfrac{L}{L+R}$.

   So, when deciding where to place the next value, use RAND to select $Q \in]0, 1[$ and then place the value to the left if $Q < \dfrac{L}{L+R}$.

B.8 Selection of $f \in M(A \to B, N)$

Select $r \in R(N)$.

Since we know the number of unimodal permutations with a given mode, we can count the total number which have a mode of at least A and at most B, and then find the proportions of that total which have each of the permissible modes in the range $A \to B$. We can then set those proportions as subintervals of $[0, 1]$ and use RAND to select one of them as the mode, M. Having selected that mode, we can then proceed as with M(M,N).

B.9 Merge Blocks

Say the merge block covers columns $M, \ldots, N$, so that we know $h(1), \ldots, h(M-1)$. We reconstruct $h(M), \ldots, h(N)$ from the merge block.

Select f from the underlying set (of degree N), normalise $(f(M), \ldots, f(N))$ so that they become a distribution of degree $N - M + 1$. Use that distribution to generate the appropriate number of observations. Append those generated observations to $(h(1), \ldots, h(M-1))$ to form the given histogram, and then continue with the analysis.

Do all of the above with every selection of a distribution.

B.10 Using a distribution to generate a simulated observation

Having selected f from the underlying set of degree N, use the points $f(1), f(1) + f(2), \ldots, f(1) + \cdots + f(N-1)$ to partition [0,1] into N subintervals. Label those subintervals $1, \ldots, N$ from left to right, and use RAND to select one of them.

B.11 Expected frequency distributions and CDFs

The program reads the upper and lower limits of the interval of interest from data.txt, and partitions it into 20 subintervals by using 21 equally-spaced points. Those 21 points partition [0,1] into usually 22 subintervals (including the two, not of the same length, outside the interval of interest).

Having found $Pr(g|h)$, the program looks to see which of those 22 intervals it falls into, and increases an accumulation-register for that interval by f^h. At the end of the iterations, it normalises the contents of the accumulation-registers, to produce the expected frequency distribution. The expected CDF is the CDF of that expected frequency distribution.

C Standard analytically-solvable problems

a. The Multinomial Theorem gives the likeliness of g given h when the underlying set is singleton.

b. The Law of Succession gives the likeliness, over S(N), of g given h when $\omega(g) = 1$.

c. The Combination Theorem gives the likeliness, over S(N), of g given h when $h = \underline{0}$.

d. The Integram Theorem gives the likeliness, over S(N), of g given h when g & h are both integrams.

e. The Ranked Law of Succession gives the likeliness, over R(N), of g given h when $\omega(g) = 1$.

For details of a-d, see [1]; for details of e, see [2].

References

[1] Lovegrove,R.,(2013), 'The Fundamentals of Likelinesses', Lovegrove Mathematical Services Research Report 2013-02, London, December 2013

[2] Lovegrove,R.,(2013), 'Ranked Distributions on Finite Domains', Lovegrove Mathematical Services Research Report 2013-02, London, December 2013